

**CLAIMS:**

- 1 1. A processor-based method performed by software emulating an instruction processor,  
2 the method comprising:  
3 processing read instructions with an emulated processor executing within an  
4 emulation environment to output independent read requests via an operand interface and an  
5 op-code interface of the emulated processor;  
6 independently comparing op-code reference data and operand reference data to  
7 operands and op-codes received in response to the read requests; and  
8 recording results of the independent comparisons.
- 1 2. The method of claim 1, wherein independently comparing further comprises:  
2 storing the op-code reference data and the operand reference data within a set of data  
3 memories of the emulated instruction processor;  
4 maintaining within the emulated instruction processor an operand data pointer to  
5 address the operand reference data and an op-code pointer to address the op-code reference  
6 data; and  
7 independently accessing the operand reference data with the operand data pointer and  
8 the op-code reference data with the op-code data pointer during processing of the read  
9 instructions to verify the received op-codes and the received operands.
- 1 3. The method of claim 2, further comprising:  
2 when processing the read instructions, determining whether each of the read  
3 instructions required an operand read request or an op-code read request; and  
4 independently updating the op-code reference pointer or the operand reference pointer  
5 based on the determination.
- 1 4. The method of claim 2, wherein the read instructions form part of an instruction  
2 stream executed by the emulated instruction processor, the method further comprising:  
3 processing a flow control instruction of the instruction stream with the emulated  
4 instruction processor; and

5           upon processing the flow control instruction, synchronizing the op-code reference  
6 pointer and the operand reference pointer to respectively address a portion of the op-code  
7 reference data and a portion of the operand reference data associated with a target address of  
8 the flow control instruction.

1       5.       The method of claim 2, compiling test software to output the operand reference data,  
2 the op-code reference data, and the instruction stream.

1       6.       The method of claim 2, wherein independently comparing further comprises:  
2           latching the op-code reference data within a first latch within the emulated instruction  
3 processor; and  
4           comparing the latched op-code referenced data and the received op-codes with a  
5 comparator to produce the results.

1       7.       The method of claim 2, wherein independently comparing further comprises:  
2           latching the operand reference data within a first latch within the emulated instruction  
3 processor; and  
4           comparing the latched operand referenced data and the received operand with a  
5 comparator to produce the results.

1       8.       The method of claim 1, further comprising:  
2           storing write data within a data memory of the emulated instruction processor;  
3           maintaining within the emulated instruction processor a write data pointer to address  
4 the write data; and  
5           processing a write instruction with the emulated processor to output a write request  
6 via a data interface of the emulated processor, wherein the write request comprises a portion  
7 of the write data referenced by the write pointer.

1 9. The method of claim 1, further comprising generating a report based on the results,  
2 wherein the report identifies any of the received op-codes that do not match the op-code  
3 reference data and any of the received operands that do not match the operand reference data.

1 10. The method of claim 1, wherein recording results comprises storing addresses  
2 associated with received operands and the op-codes, copies of the received operands or op-  
3 codes, copies of the reference operands and the reference op-codes, or copies of the  
4 instruction.

1 11. The method of claim 1, wherein recording the results comprises recording the results  
2 within a register within the emulated instruction processor.

1 12. The method of claim 1, further comprising applying bit masks to at least a portion of  
2 a result of the independent comparisons of the op-code reference data and the operand  
3 reference data to the op-code and the operand received in response to the read requests.

1 13. The method of claim 1, further comprising:  
2 storing the addresses of a small number of received operands and received op-codes  
3 within a cache within the emulated instruction processor; and  
4 selectively enabling and disabling access to the cache when executing subsequent read  
5 instructions and waiting to output read requests via the operand interface and the op-code  
6 interface when the read instructions request operands and op-codes are invalidated within the  
7 cache based on a configurable option.

1 14. A processor-based system for emulating an instruction processor comprising:  
2 a computing system to provide an emulation environment; and  
3 software executing within the emulation environment to emulate an instruction  
4 processor having an operand interface and an op-code interface,  
5 wherein the software emulates the instruction processor by processing read  
6 instructions and outputting corresponding read requests on the operand interface or the op-

7 code interface, and independently comparing op-code reference data and operand reference  
8 data to operands and op-codes received from the operand interface and op-code interface in  
9 response to the read requests.

1 15. The system of claim 14, wherein the software emulates the instruction processor by  
2 recording results of the independent comparisons within a register of the emulated instruction  
3 processor.

1 16. The system of claim 14, wherein the emulated instruction processor comprises:  
2 a first data memory to store the op-code reference data; and  
3 a second data memory to store the operand reference data.

1 17. The system of claim 16, wherein the emulated instruction processor comprises:  
2 a control unit that maintains an operand data pointer to address the operand reference  
3 data within the first data memory and an op-code pointer to address the op-code reference  
4 data within the second data memory,  
5 wherein the control unit independently accesses the operand reference data with the  
6 operand data pointer and the op-code reference data with the op-code data pointer during  
7 processing of the read instructions to verify the received op-codes and the received operands.

1 18. The system of claim 17, wherein upon processing the read instructions, the control  
2 unit determines whether each of the read instructions required an operand read request or an  
3 op-code read request, and independently updates the op-code reference pointer or the  
4 operand reference pointer based on the determination.

1 19. The system of claim 17, wherein the read instructions form part of an instruction  
2 stream executed by the emulated instruction processor, and upon processing a flow control  
3 instruction of the instruction stream, the control unit synchronizes the op-code reference  
4 pointer and the operand reference pointer to respectively address a portion of the op-code

5 reference data and a portion of the operand reference data associated with a target address of  
6 the flow control instruction.

1 20. The system of claim 19, further comprising a compiler executing on the computing  
2 system to compile test software to output the operand reference data, the op-code reference  
3 data, and the instruction stream for execution by the emulated instruction processor.

1 21. The system of claim 19, wherein the emulated instruction processor further  
2 comprises:

3 a latch to latch the op-code reference data from the first data memory; and  
4 a comparator to compare the latched op-code referenced data and the received op-  
5 codes.

1 22. The system of claim 19, wherein the emulated instruction processor further  
2 comprises:

3 a latch to latch the operand reference data from the second data memory; and  
4 a comparator to compare the latched operand referenced data and the received  
5 operands.

1 23. The system of claim 14, wherein the emulated instruction processor further  
2 comprises:

3 a data memory to store write data; and  
4 a control unit to maintain a write data pointer to address the write data, wherein the  
5 control unit processes a write instruction to output a write request via a data interface of the  
6 emulated processor, and further wherein the control unit generates the write request to  
7 comprises a portion of the write data referenced by the write pointer.

1 24. The system of claim 14, further comprising emulation control software executing on  
2 the computing system to generate a report that presents/identifies any of the received op-

3 codes that do not match the op-code reference data and any of the received operands that do  
4 not match the operand reference data.

1 25. The system of claim 14, wherein the emulated instruction processor further  
2 comprises:

3 a set of memories to store bit masks; and  
4 bit masks to compare results of the comparison of the received operands and the  
5 received op-codes to the reference operands and the reference op-codes to mask portions of  
6 these comparisons.

1 26. The system of claim 14, wherein the emulated instruction processor further comprises  
2 a cache to store the addresses of the received operands and received op-codes, wherein the  
3 emulated instruction processor selectively waits to issue read requests for subsequent read  
4 instructions via the operand interface and the op-code interface until the read instructions  
5 request operands and op-codes become invalidated within this cache based on a configurable  
6 option.

1 27. A processor-based system for emulating an instruction processor comprising:  
2 compiling means for compiling test software to produce operand reference data, op-  
3 code reference data, and an instruction stream having read instructions; and  
4 emulating means for emulating an instruction processor having an operand interface  
5 and an op-code interface,

6 wherein the emulating means comprises:

7 controlling means for controlling the emulated instruction processor to  
8 process the read instructions and output corresponding read requests on the operand  
9 interface or the op-code interface,

10 receiving means for receiving operands and op-codes from the operand  
11 interface and op-code interface in response to the read requests, and

12 comparing means for independently comparing the op-code reference data and  
13 the operand reference data to the received.

1 28. The system of claim 27, wherein the emulating means further comprises:

2 a first storing means for storing the op-code reference data; and

3 a second storing means for storing the operand reference data.

1 29. The system of claim 27, wherein the controlling means comprises:

2 first referencing means for addressing the operand reference data within the first  
3 storing means; and

4 second referencing means for addressing the op-code reference data within the second  
5 storing means.

1 30. The system of claim 27, further comprising reporting means for generating a report  
2 that presents and identifies any of the received op-codes that do not match the op-code  
3 reference data and any of the received operands that do not match the operand reference data.